Abschlussprüfung Winter 2025/26
Lösungshinweise

IHK

Fachinformatiker/Fachinformatikerin
Anwendungsentwicklung
1201

2 Entwicklung und Umsetzung von Algorithmen

Teil 2 der Abschlussprüfung

Allgemeine Korrekturhinweise

Die Lösungs- und Bewertungshinweise zu den einzelnen Handlungsschritten sind als Korrekturhilfen zu verstehen und erheben nicht in jedem Fall Anspruch auf Vollständigkeit und Ausschließlichkeit. Neben hier beispielhaft angeführten Lösungsmöglichkeiten sind auch andere sach- und fachgerechte Lösungsalternativen bzw. Darstellungsformen mit der vorgesehenen Punktzahl zu bewerten. Der Bewertungsspielraum des Korrektors (z. B. hinsichtlich der Berücksichtigung regionaler oder branchenspezifischer Gegebenheiten) bleibt unberührt.

Zu beachten ist die unterschiedliche Dimension der Aufgabenstellung (nennen – erklären – beschreiben – erläutern usw.).

Für die Bewertung gilt folgender Punkte-Noten-Schlüssel:

Note 1	=	100 – 92 Punkte	Note 2	= unter	92 – 81 Punkte
Note 3	= unter	81 – 67 Punkte	Note 4	= unter	67 – 50 Punkte
Note 5	= unter	50 – 30 Punkte	Note 6	= unter	30 – 0 Punkte

Die Vervielfältigung, Verbreitung und öffentliche Wiedergabe der Prüfungsaufgaben und Lösungen ist nicht gestattet. Zuwiderhandlungen werden zivil- und strafrechtlich (§§ 97 ff., 106 ff. UrhG) verfolgt. – © ZPA Nord-West 2025 – Alle Rechte vorbehalten!

1. Aufgabe (30 Punkte)

a) 20 Punkte

```
List<Tisch> erstelleFreieTische (
     List<Tisch> tische,
     List<Reservierung> reservierungen,
     Date datum,
     Integer anzahlPersonen)

List<Tisch> freieTische = new List<Tisch>()

Für int i = 0; i < tische.size(); i++
     Tisch t = tische.get(i)
     Wenn anzahlPersonen <= t.getMaxPersonen()und
          anzahlPersonen >= t.getMaxPersonen()/2.0
                         // keine Integerdivision!
          Boolean frei = true
          Für int j = 0; j < reservierungen.size() und frei; j++
               Reservierung r = reservierungen.get(j)
               Wenn r.getTischNr() == t.getTischNr() und
                    r.getDatum() == datum
                    frei = false
               Ende wenn
          Ende für
          Wenn frei
               freieTische.add(t)
          Ende wenn
     Ende wenn
Ende für

Return freieTische

Ende erstelleFreieTische
```

b) 10 Punkte

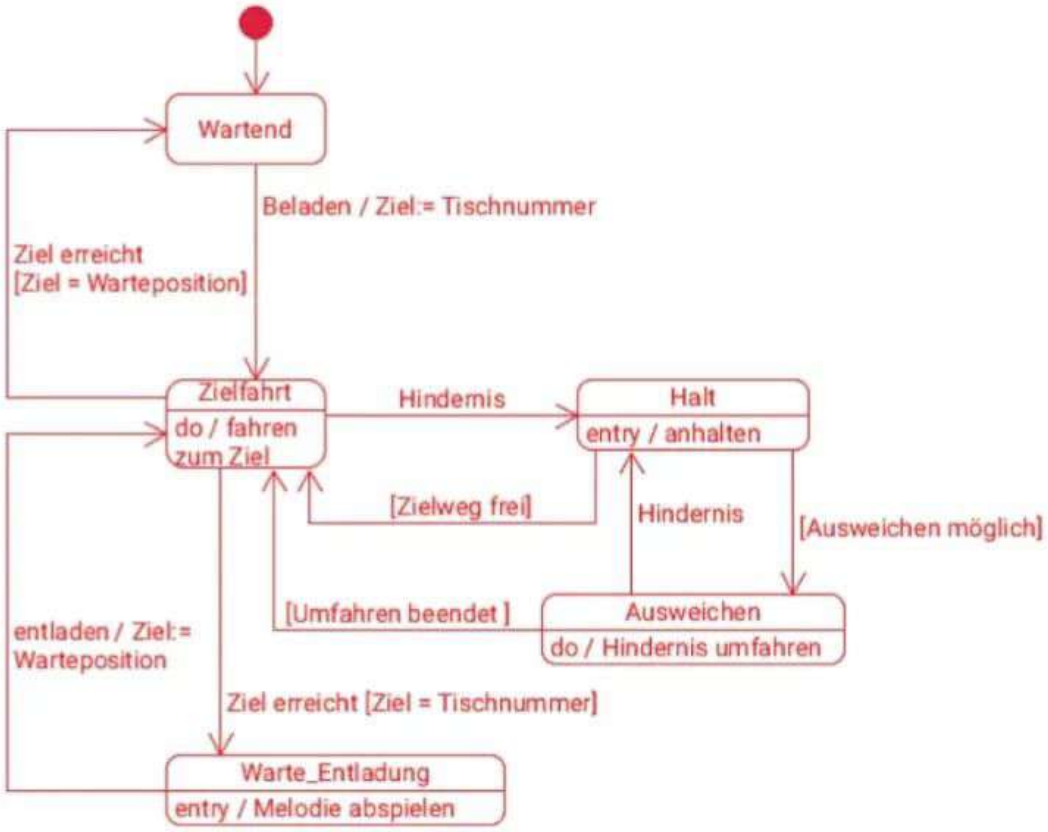
```
Tisch besterTisch(List<Tisch> freieTische, Integer anzahlPersonen)

     wenn freieTische.size() == 0
          Rückgabe null

     Tisch tisch = freieTische.get(0)
     für int i = 1; i < freieTische.size(); i++
          wenn freieTische.get(i).getPersonen() - anzahlPersonen <
               tisch.getPersonen() - anzahlPersonen
               tisch = freieTische.get(i)
          ende wenn
     end für
     Rückgabe tisch

Ende besterTisch
```

2. Aufgabe (25 Punkte)



3. Aufgabe (20 Punkte)

aa) 3 Punkte

Bei einem Unit-Test werden einzelne Teile oder Einheiten einer Anwendung, die sogenannten Units, auf ihre Funktionalität hin überprüft.

ab) 3 Punkte

Eine solche Testabdeckung beschreibt, wie viel Prozent der gesamten Anweisungen im Quellcode durch die Testfälle erreicht und ausgeführt werden, d. h. von den Tests abgedeckt sind.

ba) 10 Punkte

Testdaten	erwartetes Testergebnis	Testergebnis
int[] werte = {10, 20, 30, 5, 17};	5	0
int[] werte = {10, -10, -20, -20, 30};	-20	-20
int[] werte = {10, 18, 30, 20, -15, -25};	-25	-15
int[] werte = {-3};	-3	0
int[] werte = null;	Exception – "Array nicht auswertbar."	Exception – "Array nicht auswertbar."

bb) 4 Punkte

Position (Zeilennummer)	mögliche Korrektur
6	int kleinsterWert = werte[0];

8	for (int i = 0; i < werte.length; i++) {
	oder
	for (int i = 0; i <= werte.length - 1; i++) {

4. Aufgabe (25 Punkte)

Alternative korrekte Lösungen sind ebenfalls als richtig zu bewerten!

a) 2 Punkte

```
INSERT INTO Gerichtkategorie (GKID, Name)
VALUES (5, 'Dessert');
```

b) 5 Punkte

```
SELECT   b.BID,
         k.Name AS Kunde,
         SUM(g.Preis * bd.Menge) AS Gesamtpreis
FROM  Bestellungen b
JOIN  Bestelldetails bd ON b.BID = bd.BID
JOIN  Gerichte g ON bd.GID = g.GID
JOIN  Kunden k ON b.KID = k.KID
GROUP BY b.BID
ORDER BY b.BID;
```

c) 5 Punkte

```
SELECT k.KID, k.Name, COUNT(b.BID) AS AnzahlBestellungen
FROM Kunden k
JOIN Bestellungen b ON k.KID = b.KID
GROUP BY k.KID, k.Name
HAVING COUNT(b.BID) >= 3;
```

d) 8 Punkte

```
ALTER TABLE Kunden
ADD COLUMN Vorname VARCHAR(50),
ADD COLUMN Nachname VARCHAR(50);

UPDATE Kunden
SET  Vorname = SUBSTRING(Name, ' ', 1),
     Nachname = SUBSTRING(Name, ' ', -1);

ALTER TABLE Kunden
DROP COLUMN Name;
```

e) 5 Punkte

```
SELECT   gk.Name AS KategorieName,
         SUM(bd.Menge) AS Verkaufsanzahl
FROM Bestelldetails bd
JOIN Gerichte g ON bd.GID = g.GID
JOIN Gerichtkategorie gk ON g.GKID = gk.GKID
GROUP BY gk.Name
ORDER BY Verkaufsanzahl DESC;
```